This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method of analyzing program execution within an operating system of a multithreaded environment, comprising:

accumulating diagnostic data pertaining to a thread accessing a resource, the execution of a thread being predicated upon the thread's access to the resource within the multithreaded environment; and

storing the diagnostic data within a data structure at a location in the data structure correlated to the resource;

detecting a locking occurrence associated with the thread;

calculating a time increment corresponding to a duration that the thread remains locked; and

using the time increment to analyze the program execution.

2. (Previously Presented) The method according to claim 1, wherein the diagnostic data includes data selected from at least one of: a time measurement, program code executed by the thread, an invocation stack, and pointer data.

3. (Original) The method according to claim 1, wherein the data structure comprises a hash bucket.

4. (Original) The method according to claim 1, further comprising determining the resource.

5. (Original) The method according to claim 4, wherein determining the resource includes reading contents of a task dispatcher.

6. (Original) The method according to claim 1, further comprising storing information identifying the resource.

7. (Original) The method according to claim 1, further comprising matching an identifier corresponding to the resource to a correlative identifier corresponding to the data structure.

8. (Original) The method according to claim 7, further comprising reassigning the identifier to a second resource.

9. (Original) The method according to claim 7, further comprising assigning the correlative identifier to the data structure.

10. and 11. (Canceled)

12. (Original) The method according to claim 11, further comprising storing the time increment within the data structure.

13. (Original) The method according to claim 10, further comprising recording the time corresponding to the locking occurrence.

14. (Currently Amended) The method according to claim 1, further comprising detecting a removal of the locking occurrence.

15. (Currently Amended) The method according to claim 14, further comprising recording a time instance corresponding to the removal of the locking occurrence.

16. (Original) The method according to claim 10, further comprising recording program data relating to code executed by the thread prior to the locking occurrence.

17. (Original) The method according to claim 16, further comprising retrieving the program data from an invocation stack.

18. (Original) The method according to claim 1, further comprising displaying the diagnostic data.

19. (Currently Amended) A method of analyzing program execution within a computer system having a plurality of threads accessing a plurality of resources, comprising:

detecting a locking occurrence associated with a thread of the plurality of threads;

calculating a time increment reflective of a duration ~~a~~ the thread ~~of the plurality of threads~~ waits for access to a resource of the plurality of resources, the execution of the thread being predicated upon the thread's access to the resource; ~~and~~

storing the time increment within a hash bucket of a plurality of hash buckets comprising a hash array, each hash bucket being correlated to the resource; and

using the time increment to analyze the program execution.

20. (Previously Presented) The method according to claim 19, further comprising reallocating the plurality of resources to the plurality of hash buckets to group the diagnostic data with a different scheme.

21. (Currently Amended) An apparatus comprising:

at least one processor configured to execute a plurality of threads;

a memory; and

program code resident in the memory and configured to execute on the at least one processor, the program code configured to accumulate diagnostic data pertaining to a thread accessing a resource, the execution of a thread being predicated upon the thread's access to the resource, ~~and~~ to store the diagnostic data within a data structure at a location in the data structure correlated to the resource, wherein the program code initiates a detection of a locking occurrence associated with the thread and calculates a time

increment corresponding to a duration that the thread remains locked, and facilitates use of the time increment to analyze the program execution.

22. (Previously Presented) The apparatus according to claim 21, wherein the diagnostic data includes data selected from a group consisting of at least one of: a time measurement, program code executed by the thread, an invocation stack and pointer data.

23. (Currently Amended) The apparatus according to claim 21, wherein the data structure ~~lock of memory~~ comprises a hash bucket.

24. (Original) The apparatus according to claim 21, wherein the program code initiates a determination of the resource.

25. (Original) The apparatus according to claim 21, wherein the program code initiates storing information identifying the resource.

26. (Original) The apparatus according to claim 21, further comprising matching an identifier corresponding to the resource to a correlative identifier corresponding to the data structure.

27. (Original) The apparatus according to claim 26, wherein the program code initiates reassigning the identifier to a second resource.

28. (Original) The apparatus according to claim 26, wherein the program code initiates assigning the correlative identifier to the data structure.

29. and 30. Canceled

31. (Original) The apparatus according to claim 30, wherein the program code initiates storing the time increment within the data structure.

32. (Currently Amended) The apparatus according to claim 21, wherein the program code initiates recording a time corresponding to the [a] locking occurrence.

33. (Currently Amended) The apparatus according to claim 21, wherein the program code initiates detecting a removal of the locking occurrence.

34. (Currently Amended) The apparatus according to claim 33, wherein the program code initiates recording a time instance corresponding to the removal of the locking occurrence.

35. (Currently Amended) The apparatus according to claim 2[9]1, wherein the program code initiates recording program data relating to code executed by the thread prior to [a] the locking occurrence.

36. (Original) The apparatus according to claim 35, wherein the program code initiates retrieval of the program data from an invocation stack.

37. (Original) The apparatus according to claim 21, wherein the program code initiates a display of the diagnostic data.

38. (Currently Amended) A program product, comprising:

program code executable by a computer for analyzing program execution within an operating system of a multithreaded environment, wherein the program code is configured to accumulate diagnostic data pertaining to a thread accessing a resource, the execution of a thread being predicated upon the thread's access to the resource, ~~and~~ to store the diagnostic data within a block of the memory correlated to the resource, wherein the program code initiates a detection of a locking occurrence associated with the thread and calculates a time increment corresponding to a duration that the thread remains locked, and facilitates use of the time increment to analyze the program execution; and

a recordable medium bearing the program code.

39. (Cancelled)